DAT: Training Deep Networks Robust to Label-Noise by Matching the Feature Distributions

Yuntao Qu[1]*, Shasha Mo[1]†, Jianwei Niu[2,3,4]
[1]School of Cyber Science and Technology, Beihang University
[2]State Key Laboratory of Virtual Reality Technology and Systems, School of Computer Science and Engineering, Beihang University
[3]Hangzhou Innovation Institute of Beihang University
[4]Zhengzhou University Research Institute of Industrial Technology, Zhengzhou University
{tyqnn, moshasha, niujianwei}@buaa.edu.cn

Abstract

In real application scenarios, the performance of deep networks may be degraded when the dataset contains noisy labels. Existing methods for learning with noisy labels are limited by two aspects. Firstly, methods based on the noise probability modeling can only be applied to class-level noisy labels. Secondly, others based on the memorization effect outperform in synthetic noise but get weak promotion in real-world noisy datasets. To solve these problems, this paper proposes a novel label-noise robust method named Discrepant Adversarial Training (DAT). The DAT method has ability of enforcing prominent feature extraction by matching feature distribution between clean and noisy data. Therefore, under the noise-free feature representation, the deep network can simply output the correct result. To better capture the divergence between the noisy and clean distribution, a new metric is designed to change the distribution divergence into computable. By minimizing the proposed metric with a min-max training of discrepancy on classifiers and generators, DAT can match noisy data to clean data in the feature space. To the best of our knowledge, DAT is the first to address the noisy label problem from the perspective of the feature distribution. Experiments on synthetic and real-world noisy datasets demonstrate that DAT can consistently outperform other state-of-the-art methods. Codes are available at https://github.com/Tyqnn0323/DAT.

1. Introduction

Benefitting from the support of large-scale annotation datasets like ImageNet, deep networks have achieved eyeball-popping performance on various vision problems such as image classification, object detection and semantic segmentation, *etc*. However, annotating data is an expensive and time-consuming task, especially in manual execution. Collecting labels from crowdsourcing or crawling websites is a substitutable scheme to make annotation faster and cheaper. Inevitably, noisy labels will be introduced in the process of these low-quality-annotations. As more complicated network structures are designed for obtaining stronger fitting ability, the deep network has higher capacity to overfit noisy labels. It raises an urgent demand on investigating robust learning methods against noisy labels.

Existing methods focus on modeling noise probability or obtaining clean labels by the memorization effect. Noise modeling methods require a precondition, named the conditional independent assumption (*i.e*., noisy labels are only related to ground-true labels, but not to the data samples) [1, 2, 3, 4]. This conditional independent assumption leads such methods to only apply to class-level noisy labels. The memorization effect means that deep networks learn clean samples in a simple pattern first and then gradually learn other noisy samples [5]. Based on this fact, such methods filter noisy samples or correct noisy labels with outputs of network [6, 7, 8, 9]. Nevertheless, Jiang *et al*. found that the memorization effect will lose efficacy on real-world noisy dataset, for the reason that the real-world noisy distribution is close to the original distribution [10]. Research efforts in training with real-world noisy data based on the memorization effect achieve only marginally effective.

This paper provides a distinctive perspective that noisy labels are processed in the feature distribution instead of label distribution. To achieve this promising methodology, a novel method, named the Discrepant Adversarial Training (DAT), is proposed in this paper. DAT stands out from existing approaches in two aspects. Firstly, DAT avoids the conditional independent assumption and focuses on the feature distribution rather than the label distribution. As a result,

*The first two authors contributed equally.
†corresponding author.

it can deal with both class-level and instance-level noisy labels. To our knowledge, no prior work has addressed the label-noise in the feature space. Secondly, even if the noisy distribution is close to the original distribution, DAT can still capture the difference in feature space by calculating distribution divergence. DAT achieves the state-of-the-art accuracy on both synthetic noisy datasets with class-level noisy labels and real-world noisy datasets including instance-level noisy labels.

Contributions of this paper can be summarized as follows:

- This paper theoretically proved that matching the feature distribution from noisy to clean data can deal with the label-noise problem. To apply the theory of matching the feature distributions, the proposed DAT method performs an adversarial training on discrepancy between the classifier and the generator with the aid of auxiliary clean dataset. By the adversarial training, DAT can enforce prominent feature extraction of generator, so that a classifier even with basic structure can effortlessly output the correct result with such a high-quality generator.
- A novel metric $h\triangle\mathcal{H}$-divergence is proposed for calculating the distribution divergence. Compared with other metrics [11], the proposed metric has a tighter generalized upper bound for the problem of handling noisy labels.
- DAT can be regarded as a regularization method to prevent overfitting noisy labels, and the effect is still significant without clean data. In other words, this paper extends DAT to the absence of auxiliary clean data by a trick. The trick prevents overfitting noisy features by matching the macroscopic feature distribution from sampled subset of untrained instances.

2. Related work

2.1. Learning on noisy data

Most methods improve the robustness of deep neural networks by modeling the noise probability with a noise transition matrix. The real noise transition matrix is usually unknown, many methods are proposed to estimate it from the noisy dataset or an additional clean subset [1, 2, 4, 12]. Sukhbaatar *et al*. first employs the noise transition matrix by adding a linear layer [1]. Patrini *et al*. extends Sukhbaatar's work by applying the noise transition matrix to loss correction [2]. However, methods based on the noise transition can only deal with class-level noise due to the conditional independent assumption.

Another type of label-noise robust method attempts to filter or correct noisy data based on the memorization effect [5]. A typical approach is co-teaching that maintains two identical networks to filter noisy data by selecting a subset of small-loss samples from parallel network [6]. But filtering could lead to insufficient training, several methods are presented to solve this problem by correcting noisy labels through early network outputs rather than filtering noisy samples [8, 9, 7]. Nonetheless, Jiang et al. observed that the memorization effect does not apply to real world noisy dataset [10]. Based on this fact, the above methods only obtain outstanding results on synthetic noisy datasets, while they are difficult to achieve excellent results on real-world noisy datasets.

2.2. Feature distribution matching

Existing methods of matching the feature distribution is mainly used for unsupervised domain adaptation. Based on the theory proposed by [11], which defines two metric of distribution divergence ($\mathcal{H}$-divergence and $\mathcal{H}\triangle\mathcal{H}$-divergence) to bound the target error , many methods seek an explicit feature space representation function that can match the source and target distributions [13, 14, 15, 16, 17]. Yaroslav *et al*. presents a domain-adversarial training method, which employs an additional domain classifier to compute the $\mathcal{H}$-divergence and obtain a generic generator by minimizing the $\mathcal{H}$-divergence [14]. Saito *et al*. introduces a more effective training method that calculates the $\mathcal{H}\triangle\mathcal{H}$-divergence by maximizing the discrepancy between two label classifiers [13]. As for the problem of handling noisy labels, the above two metrics are not applicable due to the different purpose.

Although the DAT method is motivated by the domain-adaptation methods, the only similarity is that two label classifiers are used to calculates the distribution divergence. There are fundamental differences between them. (1) DAT method uses the proposed $h\triangle\mathcal{H}$-divergence to calculate distribution divergence, while domain-adaptation methods use $\mathcal{H}\triangle\mathcal{H}$-divergence or $\mathcal{H}$-divergence. (2) The domain-adaptation method deals with two domains with different data distributions, while DAT deals with two domains with different label distributions. (3) The training process of DAT is completely different from all the domain-adaptation methods.

3. The DAT method

3.1. Preliminaries

For simplicity, a binary classification problem is considered (the same analysis type is suitable for multi-class). The clean training set is expressed as $D_c = \{(x_1, y_1), \ldots, (x_n, y_n)\} \in (\mathcal{X} \times \mathcal{Y})^n$, where $x_n \in \mathcal{X}$ denotes the image data and $y_n \in \mathcal{Y} \subseteq \{0, 1\}$ is the class label. If the training set D_c contains noisy labels, it can be re-expressed as $D_\rho = \{(x_1, \widetilde{y}_1), \ldots, (x_n, \widetilde{y}_n)\} \in (\mathcal{X} \times \widetilde{\mathcal{Y}})^n$.

Let $\mathcal{Z} \subset \mathbb{R}^d$ be the feature space, $D_c^{\mathcal{Z}}$ and $D_\rho^{\mathcal{Z}}$ represent the feature distribution of clean set and noisy set respectively. A generator is a representation function $g : \mathcal{X} \to \mathcal{Z}$ that can map instance from $\mathcal{X}$ to $\mathcal{Z}$. A classifier can be represented as $h : \mathcal{Z} \to \mathcal{Y}$, which constructs classification hyperplane on feature space. For a fixed generator g, ideal functions from features to clean labels can be defined as:

$$\hat{f}_c(\mathcal{Z}) = \mathbb{E}_{x \sim \mathrm{D}_c^{\mathcal{Z}}}[f_c(x)|g(x) = z], \tag{1}$$

where $f_c(x)$ is the latent label function from instances to clean labels, and the expectation $\mathbb{E}$ is used because there may be difference x is mapped to the same z. Based on the defined latent function $\hat{f}_c(z)$, the classifier error rate $\epsilon_c(h)$ on clean set can be defined as:

$$\epsilon_c(h) = \mathbb{E}_{z \sim \mathrm{D}_c^{\mathcal{Z}}} |\hat{f}_c(z) - h(z)|. \tag{2}$$

Parallel notations $\hat{f}_\rho(z)$ and $\epsilon_\rho(h)$ are used for the latent label function from instances to noisy labels and error rate on noisy set. The optimization goal can be expressed as minimizing the clean error rate $\epsilon_c(h)$ without information about the labels of clean training set D_c. Minimizing the clean risk $\epsilon_c(h)$ means letting classifier h fit ideal function $\hat{f}_c$. If the generator g is insufficient, the $\hat{f}_c$ will be so complex that h is difficult to fit. Therefore, in the case of classifier h simple enough, it is necessary to obtain a generator g that extracts clean features.

3.2. Generalization Bound for Learning with Noise

To prove that matching the distribution can handle label-noise, this subsection bounds the clean error rate $\epsilon_c(h)$ by summing the noisy error rate $\epsilon_\rho(h)$ and the divergence between the clean and the noisy feature distributions. By minimizing the divergence on the generator g, the clean features from noisy instances can be extracted. In this case, sufficiently simple classifier h will not overfit noisy labels, and minimizing the noisy risk $\epsilon_\rho(h)$ is equivalent to minimizing the clean risk $\epsilon_c(h)$. To obtain the divergence between the clean and the noisy feature distributions, a novel metric named $h\triangle\mathcal{H}$-divergence is proposed:

Definition 1. *Given two feature distribution $D_\rho^{\mathcal{Z}}$ and $D_c^{\mathcal{Z}}$ extracted by a fixed g, and a hypothesis class $\mathcal{H}$ which is a set of binary classifiers. Through a given classifier h, $h\triangle\mathcal{H}$-divergence between $D_\rho^{\mathcal{Z}}$ and $D_c^{\mathcal{Z}}$ is:*

$$d_{h\triangle\mathcal{H}}(D_\rho^{\mathcal{Z}}, D_c^{\mathcal{Z}}) = 2 \sup_{\grave{h} \in \mathcal{H}} \left\{ \Pr_{z \sim D_c^{\mathcal{Z}}} \left[h(z) \neq \grave{h}(z) \right] - \Pr_{z \sim D_\rho^{\mathcal{Z}}} \left[h(z) \neq \grave{h}(z) \right] \right\}. \tag{3}$$

$\Pr\left[h(z) \neq \grave{h}(z)\right]$ is the probability that two classifiers output different results. $h\triangle\mathcal{H}$-divergence is similar to $\mathcal{H}\triangle\mathcal{H}$-divergence in form, but significantly different in content. $\mathcal{H}\triangle\mathcal{H}$-divergence takes absolute upper bound on the difference between two expectations, whereas $h\triangle\mathcal{H}$-divergence takes the difference directly rather than the absolute value. In addition, both two classifiers h and $\grave{h}$ in $\mathcal{H}\triangle\mathcal{H}$-divergence are arbitrary classifiers taken in the hypothetical class $\mathcal{H}$, while $h\triangle\mathcal{H}$-divergence fixes one of the classifiers. When searching for a generalized upper bound for the clean risk $\epsilon_c(h)$, the above two changes make $h\triangle\mathcal{H}$-divergence obtain a tighter upper bound of the clean risk $\epsilon_c(h)$ than $\mathcal{H}\triangle\mathcal{H}$-divergence [1]. Therefore, lower risk on clean sets can be achieved by $h\triangle\mathcal{H}$-divergence.

Through the proposed $h\triangle\mathcal{H}$-divergence, a theorem can be presented to bound the clean error rate $\epsilon_c(h)$:

Theorem 1. *Let g be a fixed representation function from $\mathcal{X}$ to $\mathcal{Z}$, $\mathcal{H}$ be the hypothesis class of Vapink-Chervonenkis dimension d. If random noisy samples of size m is generated by applying g from D_ρ-i.i.d., then with probability at least $1-\delta$, the generalized bound of the clean risk $\epsilon_c(h)$[2]:*

$$\epsilon_c(h) \leq \epsilon_\rho^m(h) + \frac{1}{2} d_{h\triangle\mathcal{H}}\left(D_\rho^{\mathcal{Z}}, D_c^{\mathcal{Z}}\right) + \lambda, \tag{4}$$

where

$$\lambda = \epsilon_c(h^*) + \epsilon_\rho(h^*) + \sqrt{\frac{4}{m}(d \log \frac{2em}{d} + \log \frac{4}{\delta})}, \tag{5}$$

$$h^* = \mathrm{argmin}_{h \in \mathcal{H}} \, \epsilon_c(h), \tag{6}$$

$$\epsilon_\rho^m(h) = \frac{1}{m} \sum_{i=1}^{m} |\hat{f}_\rho(z) - h(z)|. \tag{7}$$

The generalized bound of the clean risk $\epsilon_c(h)$ revealing that when the distribution divergence $d_{h\triangle\mathcal{H}}(D_\rho^{\mathcal{Z}}, D_c^{\mathcal{Z}})$ and the empirical noisy risk $\epsilon_\rho^m(h)$ are low, the clean risk $\epsilon_c(h)$ will be low. Minimizing the empirical noisy risk $\epsilon_\rho^m(h)$ is very simple: typically training the generator g and the classifier h on the noisy set. An effective strategy for minimizing the distribution divergence is to find a representation function that makes the feature distribution of noisy data consistent with clean data, *i.e.*, minimizing the distribution divergence $d_{h\triangle\mathcal{H}}(D_\rho^{\mathcal{Z}}, D_c^{\mathcal{Z}})$ on the generator g. According to Theorem 1, the classifier's error on the clean set will be significantly reduced with such a generator.

3.3. Discrepant Adversarial Training (DAT)

The architecture of DAT is shown in Fig. 1, which includes two classifiers sharing one generator. An extra classifier $\grave{h}$ is added to calculate the $h\triangle\mathcal{H}$-divergence. In addition to the particular architecture, two types of losses (classification loss $\mathcal{L}_{cce}$ and discrepancy loss $\mathcal{L}_{dis}$) are involved.

[1] the detailed derivation can be found in the supplementary file.
[2] proof can be found in the supplementary file.

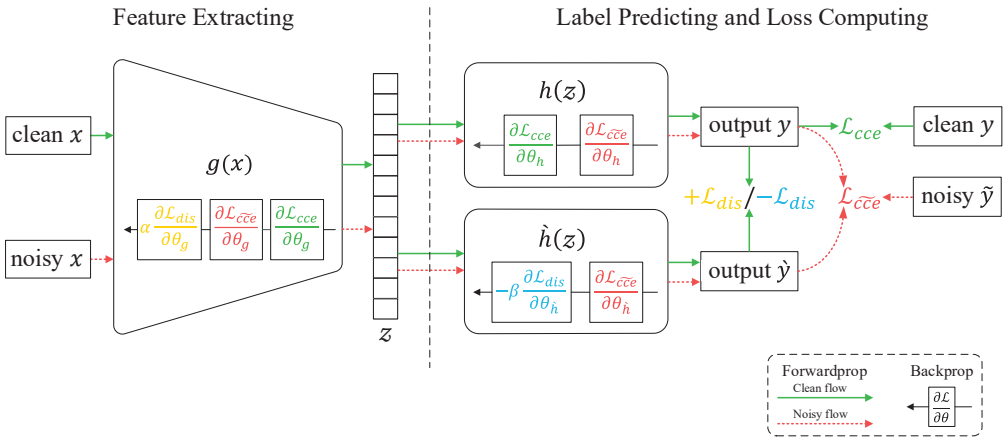

Figure 1. The architecture of DAT. One generator and two classifiers are used in DAT. In the forward pass, both noisy and clean data are used to calculate the classification loss $\mathcal{L}_{cce}$(to distinguish, $\mathcal{L}_{\widetilde{cce}}$ is the loss on noisy data) but only clean data for the discrepancy loss $\mathcal{L}_{dis}$. During backpropagation, the losses required to calculate the gradient are different for each component. For classifier h, the classification losses for clean and noisy data are used. For classifier $\grave{h}$, the classification loss for only noisy data is used, and the negative discrepancy loss is used to obtain the $h\triangle\mathcal{H}$-divergence between the noisy and clean data. For generator g, in addition to all classification losses, a positive discrepancy loss is used to minimize the $h\triangle\mathcal{H}$-divergence.

As with most previous work, the categorical cross-entropy (CCE) function is used as the classification loss:

$$\mathcal{L}_{cce} = -\frac{1}{N}\sum_{n=1}^{N}\sum_{i=1}^{k}(y_n^i \log h_n^i + y_n^i \log \grave{h}_n^i), \tag{8}$$

where N denotes the batch size and k denotes the number of classes, y_n^i is the i^{th} element of label y_n, h_n^i and $\grave{h}_n^i$ are the i^{th} element of the n^{th} classifiers' output. Although the cross-entropy loss function is sensitive to noisy labels due to its gradient property [18, 19], it is still the most commonly used loss function which offers the most efficient learning ability.

For discrepancy loss $\mathcal{L}_{dis}$, measurement such as L1 distance and Jensen-Shannon divergence (JSD) are tested. In the experiments, the L1 distance is not stable enough, while the JSD causes the generated feature distributions to be undifferentiated. Therefore, the entropy values of the respective outputs are added to the JSD in discrepancy loss $\mathcal{L}_{dis}$, making the generator output a more differentiated distribution of features. The JSD after adding entropy is:

$$\begin{aligned}\mathcal{L}_{ent} &= -\sum_{i=1}^{k}(h_n^i \log h_n^i + \grave{h}_n^i \log \grave{h}_n^i)\\ \mathcal{L}_{dis} &= \frac{1}{N}\sum_{n=1}^{N}[JSD(h_n|\grave{h}_n) + \mathcal{L}_{ent}].\end{aligned} \tag{9}$$

With all components ready, this is followed by an analysis of how the DAT relate to the theory. According to Theorem 1, our goal is to minimize the empirical noisy risk $\epsilon_\rho^m(h)$ and the distribution divergence $d_{h\triangle\mathcal{H}}(D_\rho^{\mathcal{Z}}, D_c^{\mathcal{Z}})$. As aforementioned, training with the classification loss $\mathcal{L}_{\widetilde{cce}}$ on the noisy set can directly minimize the empirical noisy risk $\epsilon_\rho^m(h)$. In contrast, minimizing the distribution divergence $d_{h\triangle\mathcal{H}}(D_\rho^{\mathcal{Z}}, D_c^{\mathcal{Z}})$ is much more sophisticated. A simple way to calculate the distribution divergence $d_{h\triangle\mathcal{H}}(D_\rho^{\mathcal{Z}}, D_c^{\mathcal{Z}})$ is making the extra classifier $\grave{h}$ output different predictions from classifier h on the clean set while output same predictions on the noisy set. The obtained divergence is then minimized on the generator g. The detailed training process of DAT can be divided into four parts:

Part A The aim of this part is to minimize the empirical noisy risk $\epsilon_\rho^m(h)$. The noisy classification loss $\mathcal{L}_{\widetilde{cce}}$ is minimized on both generator g and classifier h. Except to minimize the empirical noisy risk, the generator's feature extraction capability is coarsely trained in this part. Part A corresponds to the generator g's and classifier h's gradients back propagated by the $\mathcal{L}_{\widetilde{cce}}$ in Fig. 1.

Part B The aim of this part is to minimize the clean risk $\epsilon_c(h)$. If an auxiliary clean data set exists, the clean classification loss $\mathcal{L}_{cce}$ is minimized on both generator g and classifier h. This step is optional but can be a significant enhancement, an auxiliary clean set is used to directly reduce $\epsilon_c(h)$. Part B corresponds to the generator g's and classifier h's gradients back propagated by the $\mathcal{L}_{cce}$ backprop in Fig. 1.

Part C The aim of this part is to calculate the distribution divergence $d_{h\triangle\mathcal{H}}(D_\rho^{\mathcal{Z}}, D_c^{\mathcal{Z}})$. As previously mentioned, calculating $d_{h\triangle\mathcal{H}}(D_\rho^{\mathcal{Z}}, D_c^{\mathcal{Z}})$ requires minimizing the outputs' discrepancy on the noisy set and maximizing the outputs' discrepancy on the clean set. The discrepancy on the noisy set is minimized by minimizing the noisy classification $\mathcal{L}_{\widetilde{cce}}$

on the extra classifier $\grave{h}$. Since the noisy classification $\mathcal{L}_{\widetilde{cce}}$ on classifier h is minimized in Part A, training two classifiers with the same goal is equivalent to making their output consistent. The discrepancy on the clean set is maximized by maximizing the discrepancy loss $\mathcal{L}_{dis}$ on the extra classifier $\grave{h}$. Part C corresponds to the classifier $\grave{h}$'s gradients back propagated by the $\mathcal{L}_{\widetilde{cce}}$ and $-\mathcal{L}_{dis}$ in Fig. 1.

Part D The aim of this part is to minimize the distribution divergence $d_{h\triangle\mathcal{H}}(D_\rho^{\mathcal{Z}}, D_c^{\mathcal{Z}})$. According to the Definition 1, the probability $\Pr_{z\sim \mathrm{D}_\rho^{\mathcal{Z}}}\left[h(z) \neq \grave{h}(z)\right]$ is essentially zero after minimizing the noisy classification $\mathcal{L}_{\widetilde{cce}}$ on both classifier $\grave{h}$ and h, $h\triangle\mathcal{H}$-divergence can be directly replaced by $2\Pr_{z\sim \mathrm{D}_c^{\mathcal{Z}}}\left[h(z) \neq \grave{h}(z)\right]$ (the outputs' discrepancy on the clean set). This discrepancy is minimized on the generator g in this part. The implementation process is the minimization of the discrepancy loss $\mathcal{L}_{dis}$ on g. Part D corresponds to the generator g's gradients back propagated by the $+\mathcal{L}_{dis}$ in Fig. 1.

Part C and Part D can be viewed as an adversarial training step: maximizing the discrepancy on the classifiers but minimizing the discrepancy on the generator, thus the proposed training process is called Discrepant Adversarial Training (DAT). These four training parts are not performed separately but repeated alternately in each iteration. Two hyperparameters α and β are used to control the adversarial training. The algorithmic description of DAT is shown in Algorithm 1. After training, only the generator g and the classifier h are used in the predicting phase for testing.

3.4. Without clean dataset

The DAT method can curb overfitting noisy labels with the aid of extra clean dataset, while it also has strong noise robustness without clean datasets. This subsection proposes a trick that enables DAT to be applied in scenarios where extra clean datasets are not available. The proposed trick samples a subset from untrained noisy dataset instead of clean dataset. In the absence of clean labels, the training process omits **Part B** that is not necessary in overall DAT. The algorithm for detailed training steps is described in the supplementary files.

The core idea of DAT is to close the divergence between the noise feature distribution and the clean feature distribution by minimizing the $h\triangle\mathcal{H}$-divergence. It is worth noting that clean labels are not required in the implemention of minimizing the $h\triangle\mathcal{H}$-divergence. Moreover, the marginal distribution about the input of the noisy data is identical to the clean data, *i.e.*, $\Pr_{D_\rho}(x) = \Pr_{D_c}(x)$. Based on this fact, the extra clean data can be replaced by a subset of untrained noisy data. The main purpose of DAT without clean dataset is to prevent the generator from extracting irrelevant features from noisy instances. The key observation is that the deep network extracts additional useless detailed features of the individual noisy data in the process of overfitting the noisy labels. In this case, DAT uses the distribution of untrained data features to represent the extracted macroscopic common features and prevents the generator from extracting the noisy detailed features by minimizing $h\triangle\mathcal{H}$-divergence.

Algorithm 1 DAT-Algorithm

Input: training sets D_ρ and D_c, α and β, learning rate η, epoch T, iteration N.

Pre-treatment: divide the applied network into two parts: the network with the last few fully connected layers removed as generator g, the remaining fully connected network layers as classifier h and $\grave{h}$.

for $t = 1, 2, 3, \ldots, T$ **do**
  Shuffle training set D_ρ and D_c
  for $n = 1, 2, 3, \ldots, N$ **do**
    Fetch mini-batch $\bar{\rho}$ from D_ρ
    Fetch mini-batch $\bar{c}$ from D_c
    Calculate $\mathcal{L}_{\widetilde{cce}}$ on $\bar{\rho}$, $\mathcal{L}_{cce}$ and $\mathcal{L}_{dis}$ on $\bar{c}$
    Update $\theta_{h,\grave{h},g} = \theta_{h,\grave{h},g} - \nabla_{\theta_{h,\grave{h},g}}\mathcal{L}_{\widetilde{cce}}$
    Update $\theta_{h,g} = \theta_{h,g} - \nabla_{\theta_{h,g}}\mathcal{L}_{cce}$
    Update $\theta_{\grave{h}} = \theta_{\grave{h}} + \alpha\nabla_{\theta_{\grave{h}}}\mathcal{L}_{dis}$
    Update $\theta_g = \theta_g - \beta\nabla_{\theta_g}\mathcal{L}_{dis}$

Output: $\theta_{h,\grave{h},g}$

4. Experiments

4.1. Datasets

Synthetic Noisy Dataset All existing synthetic noisy labels are random labels generated by random algorithms. Since these algorithms do not recognize the input, but generate random labels based on the original labels with a certain probability, the synthetic label noise is all class-level noise. With a given noise rate ρ, synthetic label noise (class-level noise) can be divided into two forms: (1) Symmetrical noise: labels are flipped to other classes with the probability ρ, and the probability ρ spread uniformly among all the other classes. (2) Asymmetrcial noise: labels from a class are flipped to anther specific class with the probability ρ.

Two datasets, including MNIST [20] and CIFAR-10 [21], are corrupted as synthetic noisy dataset as with most works [2, 18, 3, 22, 8]. MINIST contains 70k of 28x28 grayscale image from 10 classes, while CIFAR-10 contains 60k of 10 objects resized to 32x32 color images. The labels in the training sets are corrupted to simulate the noisy datasets. Both types of synthetic noise are used to corrupt the training sets.

Real-world Dataset Clothing1M [23] is a large-scale real-world noisy dataset and contains more than one million images of clothes from online shopping websites. The

Table 1. Hyperparameters of DAT.

Param.	MNIST	CIFAR-10	Clothing1M	Noisy-MISC
α	$0.1 \sim 0.2$	$0.005 \sim 0.1$	0.2	$0.15 \sim 0.3$
β	$30 \sim 40$	$30 \sim 60$	60	$30 \sim 60$

labels of it are 14 classes which are generated from the text introduced by sellers. Since the crawled labels have not been reviewed, these labels are not credible and the noise level is unknown. Clothing1M provides additional validation and test sets with 14k and 10k clean data respectively. The noisy images in Clothing1M have a high degree of similarity to the true positive images, which means that most of the noisy labels are instance-level.

The noisy Mini-ImageNet and Stanford Cars (Noisy-MISC) is a benchmark dataset built by adding noisy data to Mini-ImageNet and Stanford Cars [10]. It has about 800k labels on 212,588 web images, where 12,629 web images are mislabeled in Stanford Cars and 54,400 in Mini-ImageNet. The noise levels in the Noisy-MISC is controllable. Similar to symmetric noise, the noise is uniformly distributed across the categories in the Noisy-MISC dataset.

4.2. Implementation details

Baselines for comparison DAT is compared with the following approaches: (1) F-correction [3], which applies the noise transition matrix to loss correction; (2) Co-teaching [6], which trains two parallel networks for filtering noisy samples; (3) Probabilistic End-to-end Noise Correction (PENCIL) [9], which is a current state-of-the-art method using early outputs to correct noisy labels with soft label mechanism; (4) Xiao et al. [23], which trains network with an extra clean set; (5) Categorical Cross-Entropy (CCE), which is a standard training strategy that only uses categorical cross-entropy loss.

Training Settings For the corrupted MNIST and CIFAR-10, the network structure described in [6] is used. The Adam optimizer (momentum=0.9) is applied with an initial learning rate of 0.001, and a total of 200 epochs are run with a batch size of 128. For Clothing1M, the ResNet-50 [24] is used. As the same training setting in [25], the SGD optimizer (momentum=0.9) is applied with a weight decay of 0.001 and a learning rate of 1.0×10^{-6}, and a total of 10 epochs are run with a batch size of 256. For Noisy-MISC, Inception-ResNet-v2 [26] is used. RMSProp optimizer (learning rate=0.045, epsilon=1.0) is applied with a dropout rate of 0.8, and a total of 5 epochs are run with a batch size of 32. For fair comparison, all the methods use the same network structure and parameter setup as above. The hyperparameters of DAT are shown in Table 1. For controllable noisy datasets, α and β should increase as the noise rate increases. And β is relatively sensitive and needs to be determined by the overfitting or underfitting states of the validation set.

Table 2. Test accuracy on Clothing1M.

Method	Accuracy (best/last)
CCE	70.2%/65.4%
F-correction	70.5%/67.2%
Co-teaching	71.3%/70.8%
PENCIL	71.8%/71.2%
DAT	**74.5%/73.0%**
Tong Xiao et al. * [23]	76.8%/73.5%
DAT *	**78.7%/78.0%**

4.3. Classification results

To compare DAT with other methods on synthetic noisy datasets, experiments are conducted on the corrupted MNIST and CIFAR-10. All the experiments are performed without auxiliary clean training sets, and the test accuracy of last epoch on the above two datasets is presented in Fig. 2. The line in Fig. 2 is the average result of 5 trials. Fig. 2 (a) and (c) are results conducted in symmetric noise environment, and Fig. 2 (b) and (d) are results conducted in asymmetric noise environment. It can be seen from Fig. 2 that the accuracy of DAT decreases rarely as the noise rate increases. F-correction is less effective because it does not explicitly handle the noisy labels and overfits noisy labels in the last epoch. Compared with DAT, the most competitive is PENCIL. However, PENCIL totally fails in the extremely noisy case. In the failed experiment of PENCIL, the model has misrecognized some of the true labels as the noisy labels, which leads to the mistake of modifying true labels into wrong ones. DAT almost outperforms all other methods in each experiment.

To preliminary verify the effectiveness of DAT in real-world noise cases, experiments are conducted on Clothing1M. The test accuracy of each comparison method is shown in Table 2, where * marks the method trained using extra clean set. In the first five rows of Table 2, all methods are trained without extra clean set, and the last two rows are trained with 47,570 additional clean labels. It can be seen that F-correction barely improves over CCE. This phenomenon matches its theory: conditional independent assumption only deals with class-level noise. As a distribution matching method, DAT obviously outperforms other methods with or without auxiliary clean training sets.

Finally, Noisy-MISC is used to fully illustrate the effectiveness of DAT for real noise processing. In this experiment, the performance of each method at different noise levels can be observed. Table 3 shows the results with three noise levels (10%, 30% and 50%), and all methods do not use an additional clean training set. As shown in 3, DAT consistently outperforms other methods. Although methods based on memorization effect consider the instance-level

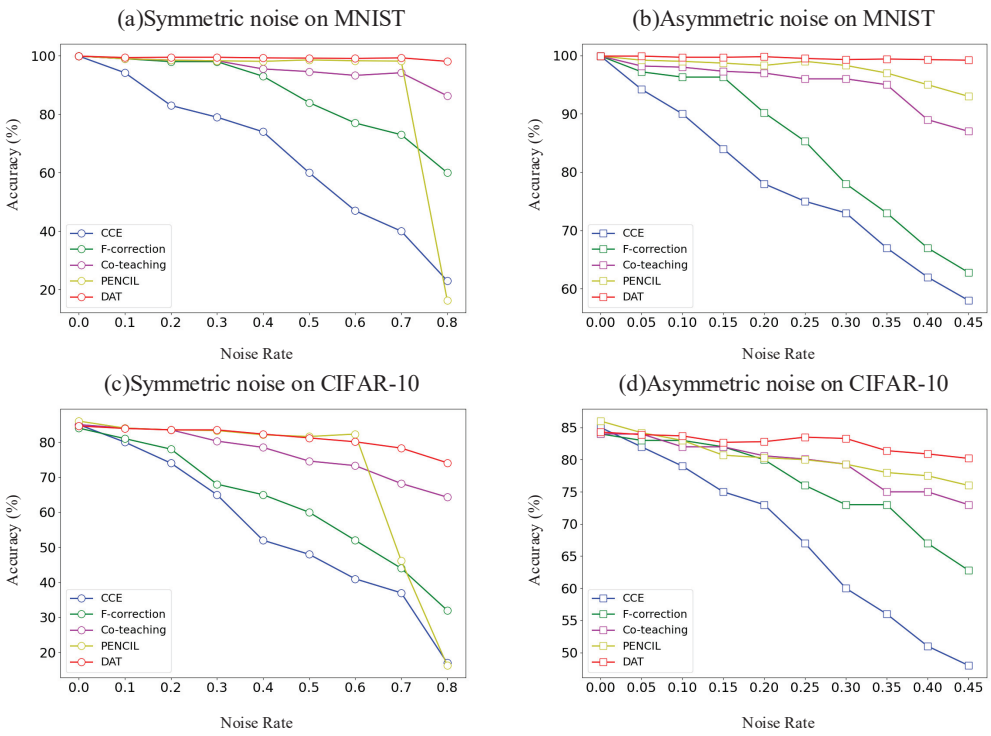

Figure 2. Test accuracy (last epoch) on MNIST and CIFAR-10.

Table 3. Test accuracy on Noisy-MISC. The results are displayed as best/last.

Dataset	Nise Rate	CCE	F-correction	Co-teaching	PENCIL	DAT
Mini-ImageNet	10%	70.2%/62.4%	70.2%/64.8%	71.6%/69.1%	72.8%/70.5%	**73.2%/72.7%**
	30%	65.7%/56.5%	66.7%/60.9%	68.7%/65.1%	69.3%/67.1%	**71.2%/70.1%**
	50%	61.5%/50.7%	63.2%/54.4%	63.7%/60.4%	65.2%/64.0%	**66.3%/66.0%**
Stanford Cars	10%	88.0%/87.8%	88.2%/87.2%	87.1%/86.9%	88.7%/87.3%	**92.4%/91.9%**
	30%	80.2%/78.9%	80.8%/78.6%	83.2%/81.2%	83.0%/82.4%	**90.5%/89.8%**
	50%	74.3%/70.6%	74.8%/71.3%	76.2%/75.6%	76.9%/76.0%	**84.4%/84.3%**

noise, they ignore the case that the noise distribution is close to the original distribution. This is particularly evident in the noisy Stanford Cars dataset, where the best result is similar to the last result, implying the smaller distributional differences. In this dataset, the improvement in DAT is more significant due to its ability to capture the small divergence in feature space.

4.4. Generator representations

To better reveal the behavior of DAT, a visualization scheme similar to [27] is used. This visualization scheme consists of the following steps: (1) Pick three categories, (2) Seek an orthogonal basis of the plane that crosses these three categories' templates, (3) Map the generator activations of the picked three categories' instance on this plane. The top 3 classes of noisy Stanford Cars are picked in this experiment, and the same training setting of Noisy-MISC in section 4.2 is chosen. Three noise levels (0%,30% and 60%) are used in this tri-categories classification experiment. For example, 60% symmetric noise means that the correct category accounts for only 40% of the data, and the other two error categories each account for 30% of the data. Both methods do not use extra training sets.

Fig. 3 shows the visualizing representations of generators trained with CCE or DAT. The first two columns represent the training set and validation set results with CCE. The first column indicates that the projections of training set are spread into the 'correct' category cluster regardless of the noise level. CCE fits all the labels the labels in the training set including the noisy labels, which means that the generator extracted unnecessary features of these noisy data and mapped them to the wrong clusters. In the second column, the feature distribution extracted by CCE overlap with each other in the validation set, verifying the conjec-

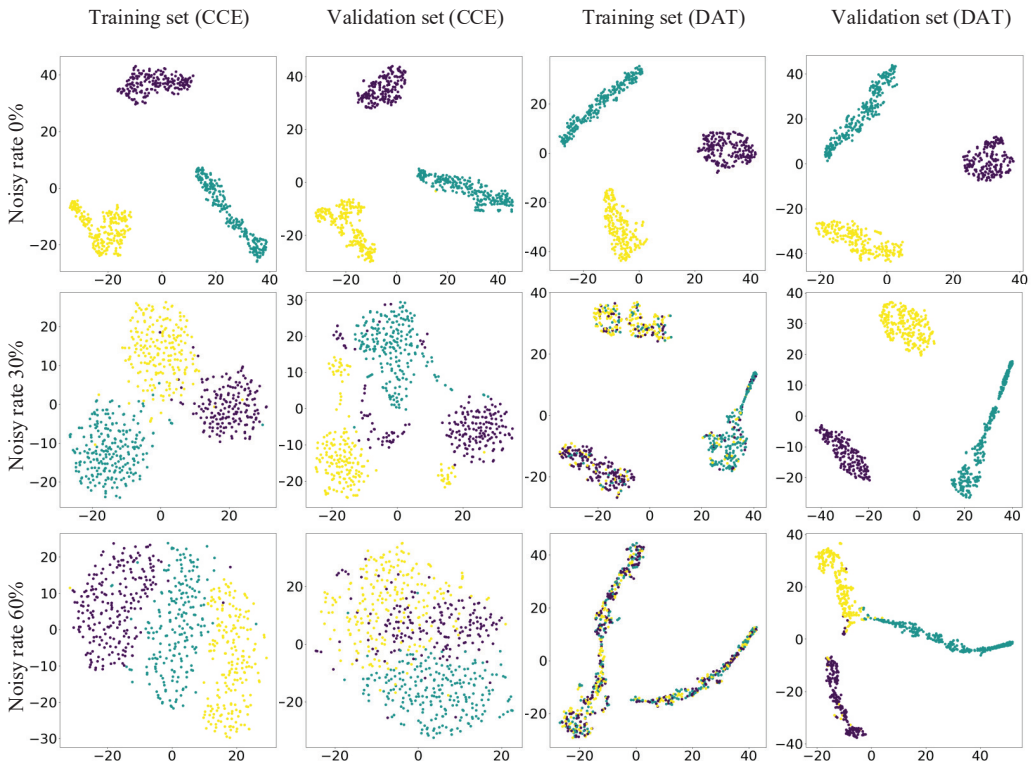

Figure 3. Visualization of the generator's activations of CCE and DAT. The dots in figure indicate the instance mapped in feature space, and colors indicate different classes. In order to demonstrate the different situations in the training set and the validation set, 600 training set instances (columns 1 and 3) and 600 validation set instances (columns 2 and 4) are visualized separately for each set of experiments. Each row shows the results of different noise levels (0%, 30% and 60%).

ture that generator extracts the unnecessary features. The last two columns represent the training set and validation set results with DAT, where the hyperparameters α is set to 0.3 and β is 30. In the noise-free environment, the distribution extracted by DAT is consistent with CCE. At 30% noise level, the generator extracts almost the same feature clusters as the noiseless training. Even at an extremely noisy environment with 60% noise level, the network still not overfits the noise with the help of DAT. As can be seen from the third column, DAT does not project the instances into the "correct" category cluster, but still clearly separates the projections into three clusters. This means that DAT does not extract the unnecessary features when the label does not match the instance features. In the last column, the feature distribution extracted by DAT is clean enough in each noise level, indicating that generator of DAT extracts more representative features. Besides, it can be observed from Fig. 3 that the category clusters of DAT lie in lines. Theoretically, when the discrepancy is minimized on the DAT generator, the feature distribution is distributed on the mid-pendant of the classifier boundary junction, making instances equally far from each classifier boundaries.

5. Conclusion

A novel method named DAT is proposed to handle noisy labels in the feature space in this paper. The process is forcing the generator to extract clean features through an adversarial training so as not to overfit the noisy labels. DAT does not need the conditional independent assumption and can deal with both class-level and instance-level noisy labels. Based on the metric $h\triangle\mathcal{H}$-divergence, DAT can capture small divergence even the noisy distribution is close to the original distribution in feature space. Consequently, DAT outperforms other method in real-world datasets. Experiments have demonstrated that the proposed DAT method has the ability of achieving the state-of-the-art result under synthetic noise and real-world noise scenarios. This research puts forward a novel point that the label-noise problem can be sovled in the feature distribution, and DAT provides a preliminary solution.

Acknowledgements This work was supported by National Natural Science Foundation of China (61772060) .

References

[1] S. Sukhbaatar, J. B. Estrach, M. Paluri, L. Bourdev, and R. Fergus, "Training convolutional networks with noisy labels," in *3rd International Conference on Learning Representations, ICLR 2015*, 2015.

[2] G. Patrini, A. Rozza, A. Krishna Menon, R. Nock, and L. Qu, "Making deep neural networks robust to label noise: A loss correction approach," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1944–1952, 2017.

[3] J. Goldberger and E. Ben-Reuven, "Training deep neural-networks using a noise adaptation layer," 2016.

[4] T. Liu and D. Tao, "Classification with noisy labels by importance reweighting," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 38, no. 3, pp. 447–461, 2015.

[5] D. Arpit, S. K. Jastrzebski, N. Ballas, D. Krueger, E. Bengio, M. S. Kanwal, T. Maharaj, A. Fischer, A. C. Courville, Y. Bengio, *et al.*, "A closer look at memorization in deep networks," in *ICML*, 2017.

[6] B. Han, Q. Yao, X. Yu, G. Niu, M. Xu, W. Hu, I. Tsang, and M. Sugiyama, "Co-teaching: Robust training of deep neural networks with extremely noisy labels," in *Advances in neural information processing systems*, pp. 8527–8537, 2018.

[7] L. Jiang, Z. Zhou, T. Leung, L.-J. Li, and L. Fei-Fei, "Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels," in *International Conference on Machine Learning*, pp. 2304–2313, 2018.

[8] D. Tanaka, D. Ikami, T. Yamasaki, and K. Aizawa, "Joint optimization framework for learning with noisy labels," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5552–5560, 2018.

[9] K. Yi and J. Wu, "Probabilistic end-to-end noise correction for learning with noisy labels," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7017–7025, 2019.

[10] L. Jiang, M. L. Di Huang, and W. Yang, "Beyond synthetic noise: Deep learning on controlled noisy labels," ICML, 2020.

[11] S. Ben-David, J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, and J. W. Vaughan, "A theory of learning from different domains," *Machine learning*, vol. 79, no. 1-2, pp. 151–175, 2010.

[12] H. Ramaswamy, C. Scott, and A. Tewari, "Mixture proportion estimation via kernel embeddings of distributions," in *International conference on machine learning*, pp. 2052–2060, 2016.

[13] K. Saito, K. Watanabe, Y. Ushiku, and T. Harada, "Maximum classifier discrepancy for unsupervised domain adaptation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3723–3732, 2018.

[14] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky, "Domain-adversarial training of neural networks," *The Journal of Machine Learning Research*, vol. 17, no. 1, pp. 2096–2030, 2016.

[15] S. J. Pan, I. W. Tsang, J. T. Kwok, and Q. Yang, "Domain adaptation via transfer component analysis," *IEEE Transactions on Neural Networks*, vol. 22, no. 2, pp. 199–210, 2010.

[16] R. Gopalan, R. Li, and R. Chellappa, "Domain adaptation for object recognition: An unsupervised approach," in *2011 international conference on computer vision*, pp. 999–1006, IEEE, 2011.

[17] M. Baktashmotlagh, M. T. Harandi, B. C. Lovell, and M. Salzmann, "Unsupervised domain adaptation by domain invariant projection," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 769–776, 2013.

[18] A. Ghosh, H. Kumar, and P. Sastry, "Robust loss functions under label noise for deep neural networks," in *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, pp. 1919–1925, 2017.

[19] Z. Zhang and M. Sabuncu, "Generalized cross entropy loss for training deep neural networks with noisy labels," in *Advances in neural information processing systems*, pp. 8778–8788, 2018.

[20] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[21] A. Krizhevsky, G. Hinton, *et al.*, "Learning multiple layers of features from tiny images," 2009.

[22] A. Vahdat, "Toward robustness against label noise in training deep discriminative neural networks," in *Advances in Neural Information Processing Systems*, pp. 5596–5605, 2017.

[23] T. Xiao, T. Xia, Y. Yang, C. Huang, and X. Wang, "Learning from massive noisy labeled data for image classification," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2691–2699, 2015.

[24] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

[25] Y. Xu, P. Cao, Y. Kong, and Y. Wang, "L_dmi: A novel information-theoretic loss function for training deep nets robust to label noise," in *Advances in Neural Information Processing Systems*, pp. 6225–6236, 2019.

[26] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2818–2826, 2016.

[27] R. Müller, S. Kornblith, and G. E. Hinton, "When does label smoothing help?," in *Advances in Neural Information Processing Systems*, pp. 4694–4703, 2019.